

Universidad Don Bosco



Ingeniería en Ciencias de la Computación (VIRTUAL)

Lenguajes Interpretados en el Servidor LIS941 G01T

Fase 1

Ing. Felipe Benjamín Acosta Coto

Nombres	Apellidos	Carné	Fotografía
Christian Gustavo	Crespin Lozano	CL060107	
Diego Guillermo	Esnard Romero	ER231474	
Diego Rene	López Martinez	LM231893	
Eduardo Ezequiel	López Rivera	LR230061	

Fecha: 30 de octubre de 2025

Índice

1. Introducción.....	2
2. Metodología del proyecto.....	3
2.1 Comparativa Scrum vs Kanban.....	3
2.2 Justificación de la metodología elegida.....	8
2.3 Aplicación práctica en la Fase #1.....	12
2.4 Cronograma de 20 días.....	15
2.4.1 Sprint 1 (Días 1 al 10) Fundación y autenticación.....	15
2.4.2 Sprint 2 (Días 11 al 20) Aprobaciones, recovery y cierre.....	16
3. Requerimientos del sistema.....	18
3.1 Requerimientos funcionales.....	18
3.2 Requerimientos no funcionales.....	21
4. Planificación y tablero de trabajo.....	23
4.1 Roles y responsabilidades.....	23
4.2 Estructura del tablero y políticas de flujo.....	24
4.3 Sprints, metas y capacidad.....	25
4.4 Pila de Producto (Product Backlog).....	25
4.5 DoR / DoD (checklists operativos).....	27
4.6 Evidencias que irán al documento.....	27
4.7 Explicación breve de uso diario.....	28
4.8 Evidencia del Backlog en Jira.....	29
5. Modelo de datos.....	30
5.1 Diagrama entidad–relación (ERD).....	30
5.2 Diccionario de datos (estructura de tablas).....	32
5.3 Reglas de integridad.....	37
6. Diseño de interfaces.....	38
7. Repositorio y control de versiones.....	44
8. Conclusiones y próximos pasos.....	45
Próximos pasos.....	45
9. Referencias.....	46

1. Introducción

El desarrollo de aplicaciones web dinámicas se ha convertido en un elemento esencial para las empresas que buscan mejorar su alcance, optimizar procesos y ofrecer servicios en línea accesibles a sus clientes. En este contexto, la asignatura Lenguajes Interpretados en el Servidor (LIS941) tiene como propósito que los estudiantes apliquen los principios del desarrollo web mediante la creación de soluciones funcionales que utilicen tecnologías modernas del lado del servidor.

El presente documento expone el trabajo correspondiente a la Fase 1 del proyecto “La Cuponera SV”, una plataforma web que permite a empresas ofrecer cupones de descuento y a los clientes adquirirlos para ahorrar en sus compras. Esta primera fase se enfoca en la implementación de los módulos base del sistema, que incluyen el registro y autenticación de usuarios, validaciones de seguridad, gestión inicial de empresas, y configuración del entorno de desarrollo.

Para la gestión del proyecto se adoptó la metodología Scrum, la cual favoreció la planificación iterativa, el trabajo colaborativo y la entrega continua de resultados mediante sprints cortos. Además, se emplearon herramientas profesionales como Jira para la organización de tareas, GitHub para el control de versiones y Laravel como framework principal para el desarrollo backend, junto con Blade y TailwindCSS en la parte visual.

La Fase 1 representa el punto de partida para la construcción completa del sistema, estableciendo la base técnica, funcional y documental que permitirá avanzar hacia las siguientes etapas del proyecto. Este documento recoge los aspectos más relevantes del proceso, desde la metodología utilizada y los requerimientos definidos, hasta los avances alcanzados en el desarrollo, las interfaces diseñadas y el repositorio del código fuente.

2. Metodología del proyecto

2.1 Comparativa Scrum vs Kanban

Tabla 1. Comparativa Scrum vs Kanban

Criterio	Scrum	Kanban	Implicaciones para Fase #1
Enfoque	Framework iterativo e incremental con timebox (cajas de tiempo) fijas por Sprint.	Método de flujo continuo; no hay iteraciones obligatorias, el trabajo pasa por columnas.	Si necesitamos hitos cerrados para la defensa, Scrum facilita objetivos por Sprint. Kanban favorece avance continuo y balance código–documento.
Estructura temporal	Sprint (1–4 semanas). Proponemos 2 sprints de 10 días o 15 días.	Sin sprints; se trabaja continuamente con lead time/cycle time.	Dos sprints calzan con 20 días y con la rúbrica (demostrables por hito).
Roles	Product Owner (PO), Scrum Master (SM), Development Team.	No define roles; se puede tener Service Delivery Manager y Product Owner opcionales.	Con 4 integrantes: 1 PO, 1 SM, 2 devs. En Kanban, menos overhead de roles.

Artefactos	Product Backlog (Pila de producto), Sprint Backlog (Pila del sprint), Increment (Incremento).	Tablero Kanban, políticas de proceso, límites WIP.	Con Scrum tendríamos Backlog priorizado por rúbrica y un Incremento demostrable por sprint.
Eventos	Sprint Planning (planificación), Daily Scrum (diaria 15'), Sprint Review (revisión/demostración), Retrospective (mejora).	No son obligatorios; reuniones ad hoc; foco en revisión del flujo y bloqueos.	Las ceremonias de Scrum ayudan a asegurar avance y calidad en 20 días.
Planificación	Se planifica al inicio del sprint; el scope se congela (salvo negociación equipo).	Priorización just in time; libertad para reordenar y replanificar a diario.	Scrum reduce cambios de alcance; Kanban permite re-priorizar si surge bloqueo técnico.
Flujo de trabajo	Empuje por sprint: tomar historias hasta cubrir la Meta del Sprint.	Pull system (sistema de tiro): se toma trabajo solo si hay capacidad en la columna siguiente.	En Kanban es natural evitar multitarea; en Scrum lo aseguramos con DoD y foco de sprint.
Límites WIP (Work In Progress)	No obligatorios (se pueden usar).	Esenciales: WIP por columna (ejemplo, <i>In Progress</i> ≤ 3).	Límites WIP reducen cuellos de botella en QA/documentación.

Priorización	PO prioriza el Product Backlog; se selecciona para el sprint según valor.	Backlog ordenado continuamente; entrada a flujo según capacidad.	En ambos, priorizamos primero mínimo funcional exigido por rúbrica.
Métricas	Velocity (velocidad), Burndown.	Lead/Cycle time, Throughput.	Para la defensa, velocity por sprint + evidencia “Done” es muy clara.
Cambios en alcance	Desaconsejados dentro del sprint (mantener foco).	Naturales y frecuentes; se reordena sin esperar fin de ciclo.	Si prevemos cambios por dependencias, Kanban es más flexible.
Visibilidad	Tablero del Sprint + artefactos; claridad de qué entra/sale del sprint.	Tablero por columnas; foco en dónde se atasca el flujo.	Ambos dan visibilidad; Kanban destaca cuellos de botella (p.ej., Code Review/QA).
Calidad	DoR/DoD formales; revisiones en Review y Retrospective.	Políticas explícitas y DoD embebido en el tablero; foco en calidad de flujo.	En Scrum, DoD obliga a cerrar validaciones (CSRF, hashing, ≥ 18 años) antes de “Done”.

Documentación y evidencias	Se planifican como historias/tareas dentro del sprint; se demuestra en Review.	Se gestionan como tarjetas en una swimlane separada (“Documentación y Evidencias”).	Ambos permiten cumplir: ≥ 5 pantallas, ERD, APA 7, repo.
Gestión de riesgos	Timeboxes evitan la “gran entrega” al final; inspección/adaptación en eventos.	Control de riesgo por WIP y visualización; se reorienta trabajo rápidamente.	Con 20 días, Scrum reduce riesgo de llegar sin demo; Kanban reacciona mejor a bloqueos.
Tamaño de equipo	Recomendado 3–9. Encaja con nuestro grupo de 4.	Cualquier tamaño; mejor con equipos pequeños para coordinar WIP.	Ambos son viables; con 4, las ceremonias de Scrum son ágiles.
Herramientas típicas	Jira/YouTrack/Azure Boards con proyectos Scrum, sprints y Burndown.	Trello/Jira con tableros Kanban, WIP y métricas de flujo.	En Jira, Scrum facilita dejar capturas de Sprint Board para la rúbrica.
Cadencia de entrega	Incrementos al final de cada sprint, listos para demo/defensa.	Entrega continua; se demuestra cuando hay lotes listos.	Para la defensa, Scrum garantiza un demo por sprint; Kanban también, pero sin hito fijo.

Ventajas clave	Foco por metas, alta previsibilidad, evidencia por hito, aprendizaje en retros.	Flexibilidad, menor overhead, optimización de flujo y WIP.	Si la fecha es fija y el mínimo funcional es crítico, Scrum brilla.
Desventajas clave	Overhead de ceremonias; cambios dentro del sprint son costosos.	Puede derivar en multitarea si no se definen políticas/WIP; falta de “deadline” interno.	Elegir según necesidad de foco vs. flexibilidad.
Cuándo conviene	Proyectos con fecha o hito fijo, alcance mínimo claro, necesidad de demo periódica.	Trabajo con prioridades variables, mezcla fuerte de tareas técnicas + documentales en paralelo.	Fase #1: defensa y rúbrica cerrada favorecen Scrum; se pueden combinar prácticas Kanban (WIP).
Aplicación práctica en este proyecto	2 sprints de 10 días: Sprint 1y Sprint 2	Flujo continuo con columnas	Cualquiera es viable; recomendación: Scrum con límites WIP (híbrido) para asegurar demo y evitar cuellos de botella.

Fuente: *Nota*. Elaboración propia

2.2 Justificación de la metodología elegida

Elegimos Scrum porque ofrece una cadencia corta y enfocada, con roles y artefactos que aseguran entregas incrementales del **mínimo funcional obligatorio** (login por roles, registros, alta de administradores, aprobación de empresas con % de comisión, recuperación de contraseña y validaciones), además de permitir evidenciar continuamente el avance que exige la rúbrica (tablero, ERD, mockups y repositorio). Nuestro equipo es de **4 integrantes**, tamaño ideal para un equipo de desarrollo ágil.

1) Artefactos de Scrum

- **Product Backlog o Pila de Producto:** lista priorizada de historias de usuario y tareas. Aquí mapeamos cada requisito de la rúbrica y RF/RNF, de mayor a menor valor para la fase.
- **Sprint Backlog o Pila del Sprint:** subconjunto del Product Backlog que comprometemos en el sprint actual, más el plan de trabajo.
- **Increment o Incremento:** resultado ejecutable y demostrable al final del sprint (por ejemplo, login y registros funcionando con validaciones y evidencias).

2) Eventos de Scrum

- **Sprint:** ciclo con duración fija (timebox). Usaremos **2 sprints** dentro de los 20 días.
- **Sprint Planning o Planificación del Sprint:** seleccionamos ítems de mayor valor (mínimo funcional + evidencias) y definimos metas claras.
- **Daily Scrum o Reunión Diaria (máx. 15 min y Reunión cada dos días):** sincronización rápida: qué hice, qué haré, qué bloquea.

- **Sprint Review o Revisión del Sprint:** demostramos el **incremento** a los interesados (profesor o equipo) y recogemos feedback.
- **Sprint Retrospective o Retrospectiva:** mejoramos el proceso (qué mantener, qué cambiar) antes del siguiente sprint.

3) Definiciones de calidad de trabajo

- **Definition of Ready (DoR) o Definición de Listo:** una historia entra al sprint solo si tiene descripción, campos/validaciones requeridos, criterios de aceptación y dependencias claras.
- **Definition of Done (DoD) o Definición de Hecho:** consideraremos “hecho” cuando exista: validación del lado servidor, mensajes de error/éxito amigables, **CSRF** activo, contraseñas **hasheadas**, pruebas manuales documentadas, commits con convención, pull request revisado, y **evidencia (capturas)** en el repo/tablero.

4) Priorización alineada a la rúbrica

Scrum nos permite ordenar el **Product Backlog** por impacto en la nota: primero funcionalidades mínimas operativas y sus validaciones; en paralelo, tareas de **documentación y evidencias** (ERD, capturas ≥ 5 , referencias APA, enlace al repo). Así garantizamos que cada **Sprint Review** muestre avances que suman directamente.

5) Gestión de riesgos en 20 días

- **Riesgo de sobrecarga final:** lo mitigamos con **dos sprints** con incrementos demostrables.

- **Riesgo de deuda técnica:** la **DoD** obliga a cerrar validaciones y seguridad básica antes de “Done”.
- **Riesgo de bloqueo individual:** la **Daily** detecta impedimentos a tiempo (migraciones, middleware, recovery flow).

6) Roles (para equipo de 4 en nuestro caso)

- **Product Owner (PO) académico:** 1 persona. Prioriza la Pila de Producto según rúbrica y aprueba criterios de aceptación.
- **Scrum Master:** 1 persona. Protege la timebox, elimina impedimentos y cuida la calidad del proceso (DoR/DoD).
- **Development Team o Equipo de Desarrollo:** 2 personas (full-stack/documentación), con ownership temporal por épicas:
 - **Auth & Seguridad** (roles, CSRF, hashing, recovery),
 - **Empresas & Administración** (registro/aprobación/% comisión),
 - **Documento/ERD/Mockups** (esta parte se reparte para que nunca quede vacía).

En nuestro caso, todos los integrantes del grupo colaboramos en QA y en completar evidencias antes de abrir nuevas tareas.

7) Métricas

- **Velocity o Velocidad:** cantidad de historias/criterios de aceptación completados por sprint (solo lo “Done” cuenta).
- **Burndown Chart → Gráfico de Quemado:** opcional para visualizar trabajo restante del sprint.

8) Plan de Sprints (resumen)

- **Sprint 1 (Días 1–10):**
 - Documento: metodología (este apartado), tablero activo, RF/RNF.
 - **ERD + migraciones/seeders** iniciales.
 - **Auth por roles** (admin/empresa/cliente).
 - **Registro de empresas y clientes + alta de administradores.**
 - **DoD:** validaciones clave (≥ 18 años), CSRF, hashing, mensajes de error, README y .env.example.
- **Sprint 2 (Días 11–20):**
 - **Aprobación/Rechazo de empresas con % de comisión.**
 - **Recuperar contraseña** (todos los roles).
 - **Mockups/capturas (≥ 5 distintas a login/registro)** y pulido de UI.
 - Cierre de documento (APA 7), enlace a repo y **ensayo de demo.**

2.3 Aplicación práctica en la Fase #1

Recordemos que escogimos la metodología Scrum y somos 4 integrantes de grupo.

Equipo:

- **Product Owner (PO):** prioriza el Product Backlog según rúbrica y valida criterios de aceptación.
- **Scrum Master (SM):** protege la timebox, aplica DoR/DoD, remueve impedimentos.
- **Dev 1–2 (Equipo de Desarrollo):** construcción full-stack + documentación. Se asignan “épicas” (Auth y Seguridad; Empresas y Administración; Documentación/ERD/Mockups). Todos apoyan en parte de QA.

Artefactos y su uso:

- **Product Backlog (Pila de Producto):** historias de usuario priorizadas:
 - EP1-Auth y Seguridad (login por roles, registro empresa/cliente, recuperación de contraseña, hashing, CSRF, validaciones).
 - EP2-Administración (alta de administradores, aprobación/rechazo empresas, % comisión).
 - EP3-Datos (ERD, migraciones/seeder).
 - EP4-Evidencias (≥ 5 pantallas, capturas del tablero, referencias APA 7, repo).
- **Sprint Backlog:** subconjunto elegido en cada sprint con tareas técnicas y documentales.
- **Incremento:** funcional mínimo ejecutable y demostrable al final de cada sprint.

Eventos:

- **Sprint Planning (Planificación):** seleccionar historias por valor de rúbrica y capacidad; definir meta del sprint y tareas.
- **Daily (15 min):** qué hice, qué haré, bloqueos; mover tarjetas en el tablero.
- **Sprint Review (Revisión):** demo del incremento (login/registro/aprobación, etc.) y evidencias.
- **Retrospective (Retro):** mejora del proceso (ajuste de WIP, pairing, pruebas).

Tablero y políticas:

- **Columns:** Backlog → To Do → In Progress → Code Review/QA → Ready for Demo → Done.
- **Límites WIP:** $In\ Progress \leq 3$, $Code\ Review/QA \leq 2$.
- **Reglas:** no mover a “Ready for Demo” sin cumplir DoD. Prioridad a vaciar “Code Review/QA” antes de tomar nuevas tareas.

DoR / DoD (checklist resumida):

- **DoR (para entrar al sprint):** historia con descripción, criterios de aceptación, dependencias claras, diseño/campos si aplica.
- **DoD (para marcar Done):** validación server-side (incluye ≥ 18 años), mensajes de error/éxito, CSRF activo, **hashing** de contraseñas, pruebas manuales documentadas, pull request revisado, evidencia (capturas) y actualización del README.

Control de versiones y ramas.

- GitHub/GitLab: ramas main, dev, feature siendo conformada por épica o tickets.
- Convención de commits: feat:, fix:, docs:, refactor:.
- **PR/MR** obligatorio con revisión cruzada (Code Review/QA).

Evidencias solicitadas por la rúbrica.

- **Jira/Trello**: capturas del Springboard.
- **Mockups** / **≥ 5 pantallas** (distintas a login/registro).
- **ERD + estructura de tablas**.
- **Repo** accesible con .env.example, migraciones/seeders y README con pasos de instalación.

2.4 Cronograma de 20 días

2.4.1 Sprint 1 (Días 1 al 10) Fundación y autenticación

Tabla 2

Cronograma del Sprint 1

Día	Actividad clave	Entregables / Evidencias
D1	Sprint Planning 1, definir roles, crear proyecto en Jira/Trello, acordar DoR/DoD	Captura del Springboard creado; Backlog priorizado
D2	Redactar RF/RNF en documento; ERD inicial; crear repo y estructura (migraciones/seeders mínimos, .env.example)	ERD v1 + diccionario; enlace al repo
D3	Autenticación por roles (admin/empresa/cliente) – scaffolding y rutas	Login básico funcionando
D4	Registro de empresas (campos requeridos + validaciones)	Form operativo + mensajes de error/éxito
D5	Registro de clientes (incluye validación ≥ 18 años)	Form operativo + validaciones
D6	Alta de administradores (solo rol admin)	Vista/acción para crear admin
D7	Endurecer seguridad: CSRF , hashing ; mensajes consistentes; actualizar README	Pruebas manuales documentadas

D8	Avance de mockups (home, panel admin); captura del Sprint Board	≥2 capturas (board + pantallas)
D9	Code Review/QA de todo el sprint; corrección de defectos	Tarjetas movidas a “Ready for Demo/Done”
D10	Sprint Review (demo: login/registro/alta admin) y Retrospective	Video/capturas de demo; notas de retro

Fuente: *Nota*. Elaboración propia

2.4.2 Sprint 2 (Días 11 al 20) Aprobaciones, recovery y cierre

Tabla 3

Cronograma del Sprint 2

Día	Actividad clave	Entregables / Evidencias
D11	Sprint Planning 2 (selección por valor); preparar tablero	Sprint Goal definido; Board actualizado
D12	Aprobación/Rechazo de empresas	Flujo de cambio de estado operativo
D13	% de comisión por empresa (reglas y validaciones)	Persistencia y validación (0–100%)

D14	Recuperar contraseña (todos los roles) – backend	Envío simulado/log; tokens/expiración
D15	Recuperar contraseña – UI/validaciones/mensajes	Flujo completo probado
D16	Mockups/Capturas (≥ 5 distintas a login/registro): home cupones, bandeja aprobaciones, detalle/compra simulada, recuperación, panel admin	Carpeta de evidencias actualizada
D17	Pulido de UI y accesibilidad; pruebas negativas (errores, accesos por rol)	Checklist de pruebas ejecutado
D18	Documento final (APA 7): metodología, RF/RNF, ERD+tablas, capturas, enlaces a tablero y repo	Versión candidata a entrega
D19	Ensayo de demo ; checklist de rúbrica; segunda captura del Springboard	Todo “Done” para mínimo funcional
D20	Sprint Review final (demo integral) y Retrospective ; entrega	Entrega oficial + evidencias finales

Fuente: *Nota*. Elaboración propia

3. Requerimientos del sistema

3.1 Requerimientos funcionales

En la tabla de Requerimientos Funcionales se incluye la marca **[F1]** para identificar el **alcance obligatorio de la Fase #1** según la rúbrica. Todo requisito marcado con **[F1]** **debe quedar implementado, probado y evidenciado** en esta entrega. Los requisitos **sin [F1]** forman parte del **alcance total del sistema** y se documentan para trazabilidad, pero su implementación completa podrá abordarse en fases posteriores.

Tabla 4

Tabla de Requerimientos Funcionales

ID	Módulo	Requisito	Descripción breve	F1	Criterios de aceptación (resumen)
RF-01	Público	Ver promociones	Visitantes ven listado de cupones disponibles (no “no disponible”).	✓	Listado visible sin login; no aparecen ofertas “no disponible”.
RF-02	Cliente	Registro de clientes	Registrar usuario, correo, contraseña, nombres, apellidos, DUI, fecha de nacimiento.	✓	Alta exitosa; errores por campos vacíos/formatos inválidos; cálculo de ≥ 18 años correcto.
RF-03	Cliente	Login cliente	Iniciar/cerrar sesión cliente.	✓	Acceso solo con credenciales válidas; mensajes claros en falla; logout limpia sesión.
RF-04	Cliente	Explorar ofertas	Filtrar/buscar por palabra, empresa, precio, vigencia.		Filtros funcionales; paginación activa.

RF-05	Cliente	Compra simulada	“Pagar” cupón con tarjeta simulada (número/vencimiento/CVV). Máx. 5 por oferta.		Validaciones de tarjeta y vigencia; límite 5 por oferta/cliente.
RF-06	Cliente	Factura y códigos	Generar factura y código único por cupón; guardar en historial.		Factura con identificador; códigos únicos; registro en BD.
RF-07	Cliente	Historial de compras	Ver cupones comprados y facturas.		Lista con fechas/estados; descarga/visualización de factura.
RF-08	Empresa	Solicitud de registro	Enviar solicitud: nombre, NIT, dirección, teléfono, correo, usuario, contraseña.	✓	Solicitud queda en estado pending; validaciones de formato; confirmación visual.
RF-09	Empresa	Perfil empresa	Editar datos (excepto NIT) tras aprobación.		Cambios persisten; restricciones por rol.
RF-10	Empresa	Publicar ofertas	Crear/editar oferta (título, precio regular/oferta, fechas inicio/fin, límite canje, cantidad opcional, descripción, estado).		Validaciones de campos; estado “disponible/no disponible”.
RF-11	Empresa	Listar/estado ofertas	Ver y cambiar estado de ofertas propias.		Cambios de estado reflejados en la web pública.
RF-12	Todos	Recuperar contraseña	Flujo de recuperación para cliente/empresa/admin.	✓	Solicitud genera token con expiración; enlace o código; reset exitoso; mensajes claros.
RF-13	Admin	Gestionar administradores	Crear usuarios con rol admin.	✓	Solo admin puede crear; nuevo admin inicia sesión con credenciales.

RF-14	Admin	Aprobar/Rechazar empresas + %	Revisar solicitudes; aprobar/rechazar y definir % comisión (0 a 100).	✓	Empresa pasa a approved/rejected; % guardado; bitácora mínima del cambio.
RF-15	Admin	Moderación de ofertas	Pausar/eliminar ofertas que incumplan políticas.		Solo admin; cambios reflejados en catálogo.
RF-16	Admin	Reportes de negocio	Totales por empresa: cupones vendidos, ventas, ganancias (según %).		Fechas filtrables; totales correctos por empresa/global.
RF-17	Todos	Autenticación por roles	Rutas/vistas protegidas para admin/empresa/cliente.	✓	Accesos indebidos redirigen a login o 403; middleware operativo.
RF-18	Todos	Validaciones y mensajes	Formularios con validación servidor y mensajes consistentes.	✓	Errores por campo; mensajes de éxito; sin trazas técnicas al usuario.
RF-19	Transversal	Auditoría básica	Registrar eventos clave (creación usuarios, aprobaciones, % comisión, logins fallidos).		Registro con usuario/fecha/acción; consultable por admin.
RF-20	Transversal	Búsqueda/filtrado	Paginación y filtros en listados (ofertas, empresas, usuarios).		Paginación configurada; filtros combinables.

Fuente: *Nota*. Elaboración propia

3.2 Requerimientos no funcionales

Tabla 5

Tabla de Requerimientos No Funcionales

ID	Categoría	Requisito	Criterios / Medidas verificables
RNF-01	Seguridad	Hash de contraseñas	Uso de bcrypt/argon2; nunca texto plano; verificación al crear/actualizar usuario.
RNF-02	Seguridad	Protección CSRF/XSS	Token CSRF en todos los formularios; escape/sanitización de salidas y entradas.
RNF-03	Seguridad	Control de acceso por rol	Middleware/guards; pruebas de acceso: rutas de admin/empresa/cliente devuelven 403 o redirigen si no corresponde.
RNF-04	Seguridad	Política de contraseñas	Mínimo 8 caracteres; validación en alta y recuperación; mensajes claros.
RNF-05	Calidad datos	Validación robusta	Tipos/longitudes/rangos; cálculo ≥ 18 años por fecha (no por año fijo); mensajes por campo.
RNF-06	Errores	Manejo de errores	Páginas 4xx/5xx personalizadas; detalles técnicos solo en logs.
RNF-07	Rendimiento	Tiempo de respuesta	CRUD típico < 1 s en entorno local; paginación habilitada.
RNF-08	Rendimiento	Paginación	Listados con 10–20 ítems por página; sin dumps masivos.
RNF-09	Mantenibilidad	Versionamiento	Git con ramas main, dev, feature/*; PR/MR con revisión cruzada.

RNF-10	Configuración	Variables de entorno	.env y .env.example; README con pasos: composer install, migraciones, seeders.
RNF-11	Arquitectura	MVC/Framework	Uso de Laravel/CodeIgniter; separación Controlador–Modelo–Vista; middleware para roles.
RNF-12	Observabilidad	Logs/Auditoría	Log de eventos clave (aprobaciones, % comisión, errores auth) con timestamp y usuario.
RNF-13	Usabilidad	Mensajería consistente	Sistema de alertas (éxito/advertencia/error) uniforme; textos comprensibles.
RNF-14	Accesibilidad	Accesibilidad básica	Etiquetas de inputs, foco/teclado, contraste suficiente; formularios navegables.
RNF-15	Compatibilidad	Navegadores	Chrome/Edge/Firefox desktop; layout responsive básico.
RNF-16	Privacidad	Minimización de datos	Exponer sólo lo necesario; restringir NIT/DUI a roles autorizados.
RNF-17	Retención	Retención mínima	Guardar únicamente datos necesarios para operación y evaluación académica.
RNF-18	Evidencia	Evidencia trazable	Jira/Trello con tarjetas movidas por estados; ≥ 5 capturas de pantallas; ERD y diccionario; referencias APA 7; enlace al repo.
RNF-19	Calidad	Criterios de aceptación	Cada historia con criterios alineados a rúbrica y DoD (Definition of Done).

Fuente: *Nota*. Elaboración propia

4. Planificación y tablero de trabajo

4.1 Roles y responsabilidades

Tabla 6

Tabla de Roles y responsabilidades

Rol	Responsable	Responsabilidades clave	Entregables tangibles
Product Owner (PO)	Integrante 1	Prioriza la Pila de Producto, valida criterios de aceptación, decide alcance de sprint	Backlog priorizado; “Aceptado/No aceptado” por historia; notas de Review
Scrum Master (SM)	Integrante 2	Facilita eventos, cuida DoR/DoD, elimina impedimentos, vigila foco	Agenda de eventos; registro de impedimentos; acuerdos de Retro
Dev A (Full-stack)	Integrante 3	Auth & Seguridad: login por roles, registros, recuperación, validaciones, CSRF/hash	Ramas/PRs de Auth; pruebas manuales; capturas de evidencia
Dev B (Full-stack)	Integrante 4	Empresas & Administración: solicitud de empresa, aprobación + % comisión, alta de admins	Ramas/PRs de Admin; migraciones/seeders; capturas de evidencia

Fuente: *Nota*. Elaboración propia

4.2 Estructura del tablero y políticas de flujo

Tabla 7

Tabla de estructura de tablero y políticas de flujo

Columna	Propósito	Regla de movimiento	Límite WIP
Backlog	Todas las historias (priorizadas por rúbrica)	Solo PO reordena	—
To Do (Sprint)	Historias del Sprint Backlog con DoR cumplida	Entra tras Planning	—
In Progress	Desarrollo activo	Una persona/tarea; pairing opcional	≤ 3
Code Review / QA	Revisión de código, pruebas, validaciones	PR abierto; otro integrante revisa	≤ 2
Ready for Demo	Cumple DoD y espera demo	Adjuntar evidencias antes de mover	—
Done	Aprobado en Review por PO	SM/PO mueven tras demo	—

Fuente: *Nota*. Elaboración propia

Reglas prácticas:

- 1) No tomar nuevas tarjetas si Code **Review/QA** está en su WIP máximo.
- 2) “Termina antes de empezar”.
- 3) Los bugs vuelven a In Progress con etiqueta **bug** y causa registrada.

4.3 Sprints, metas y capacidad

Tabla 8

Tabla de sprints, metas y capacidad

Sprint	Días	Meta del Sprint	Historias objetivo (clave)
Sprint 1	1–10	Incremento 1: autenticación por roles y registros operativos con seguridad base + ERD inicial	US-01 Login por roles [F1], US-02 Registro clientes (≥ 18) [F1], US-03 Registro empresas [F1], US-07 ERD + migraciones v1, US-11 Alta de administradores [F1]
Sprint 2	11–20	Incremento 2: aprobación de empresas con % comisión + recuperación de contraseña + evidencias de rúbrica	US-04 Aprobación/Rechazo + % [F1], US-05 Recuperar contraseña (todos) [F1], US-09 ≥ 5 pantallas (mockups/capturas), US-10 Documento final (APA 7), US-12 Capturas del Springboard

Fuente: *Nota*. Elaboración propia

4.4 Pila de Producto (Product Backlog)

Tabla 9

Tabla de product backlog

ID	Historia	Criterios de aceptación (resumen)	Marca
US-01	Como usuario quiero iniciar/cerrar sesión según mi rol para acceder solo a mis vistas	Dado credenciales válidas \rightarrow accede; inválidas \rightarrow mensaje claro; rutas protegidas por rol; logout funciona	F1
US-02	Como cliente quiero registrarme con mis datos y validación ≥ 18	Valida campos/formatos; cálculo edad por fecha; mensajes por campo; alta exitosa	F1

US-03	Como empresa quiero solicitar registro con datos de empresa	Estado inicial pending; validaciones; confirmación visual	F1
US-04	Como admin quiero aprobar/rechazar empresas y definir % comisión	Cambia a approved/rejected; % $\in [0,100]$; registra auditoría mínima	F1
US-05	Como usuario quiero recuperar mi contraseña	Solicitud genera token/expira; reset exitoso; mensajes claros	F1
US-06	Como visitante quiero ver ofertas disponibles	Solo “disponibles”; paginación; filtros básicos	
US-07	Como equipo quiero un ERD y migraciones base	ERD con tablas users, companies...; migraciones probadas	
US-08	Como empresa aprobada quiero publicar ofertas	Crear/editar con campos del enunciado; estado visible	
US-09	Como equipo quiero ≥ 5 pantallas (mockups/capturas)	Home, bandeja de aprobaciones, detalle/compra simulada, recuperación, panel admin	
US-10	Como equipo quiero el documento final en APA 7 + enlaces	Secciones completas; enlaces a repo/tablero; checklist rúbrica	
US-11	Como admin quiero crear más administradores	Solo admin; nuevo admin inicia sesión	F1
US-12	Como equipo quiero capturas del Springboard (S1 y S2)	Board con columnas y tarjetas distribuidas; una tarjeta abierta con CA y adjuntos	

Fuente: *Nota*. Elaboración propia

4.5 DoR / DoD (checklists operativos)

Tabla 10

Tabla de definition of ready and definition of done

Lista	Ítems que deben cumplirse
Definition of Ready (DoR)	Historia descrita; criterios de aceptación (Given/When/Then); dependencias claras; mockup/campos si aplica; estimada; sin bloqueos conocidos
Definition of Done (DoD)	Validación en servidor (incl. ≥ 18), CSRF activo, contraseñas hasheadas; pruebas manuales documentadas; PR/MR revisado; mensajes de error/éxito; capturas adjuntos; README actualizado si aplica

Fuente: *Nota*. Elaboración propia

4.6 Evidencias que irán al documento

Tabla 11

Tabla de Evidencias

Evidencia	Cuándo	Cómo obtenerla	Archivo sugerido (ejemplo)
Springboard (S1)	Día 8–10	Jira → Active sprints → mostrar todas las columnas, abrir una tarjeta con Criterios de Aceptación y Adjuntos → Win+Shift+S (o Cmd+Shift+4)	jira_sprint1_board_YYYYMMDD.png
Springboard (S2)	Día 18–19	Mismo procedimiento, con tarjetas finales en “Done”	jira_sprint2_board_YYYYMMDD.png

Pantallas (≥5)	Día 16–19	Capturas de Home, Bandeja de Aprobaciones, Detalle/Compra simulada, Recuperación, Panel Admin	ui_home.png, ui_aprobaciones.png, etc..
ERD	Día 2–3	Exportar PNG desde Draw.io/Lucidchart/DBDiagram	erd_v1.png o erd_final.png
Repo	Día 2 y final	URL + README con instalación; incluir .env.example, migraciones/seeder	enlace en documento

Fuente: *Nota*. Elaboración propia

4.7 Explicación breve de uso diario

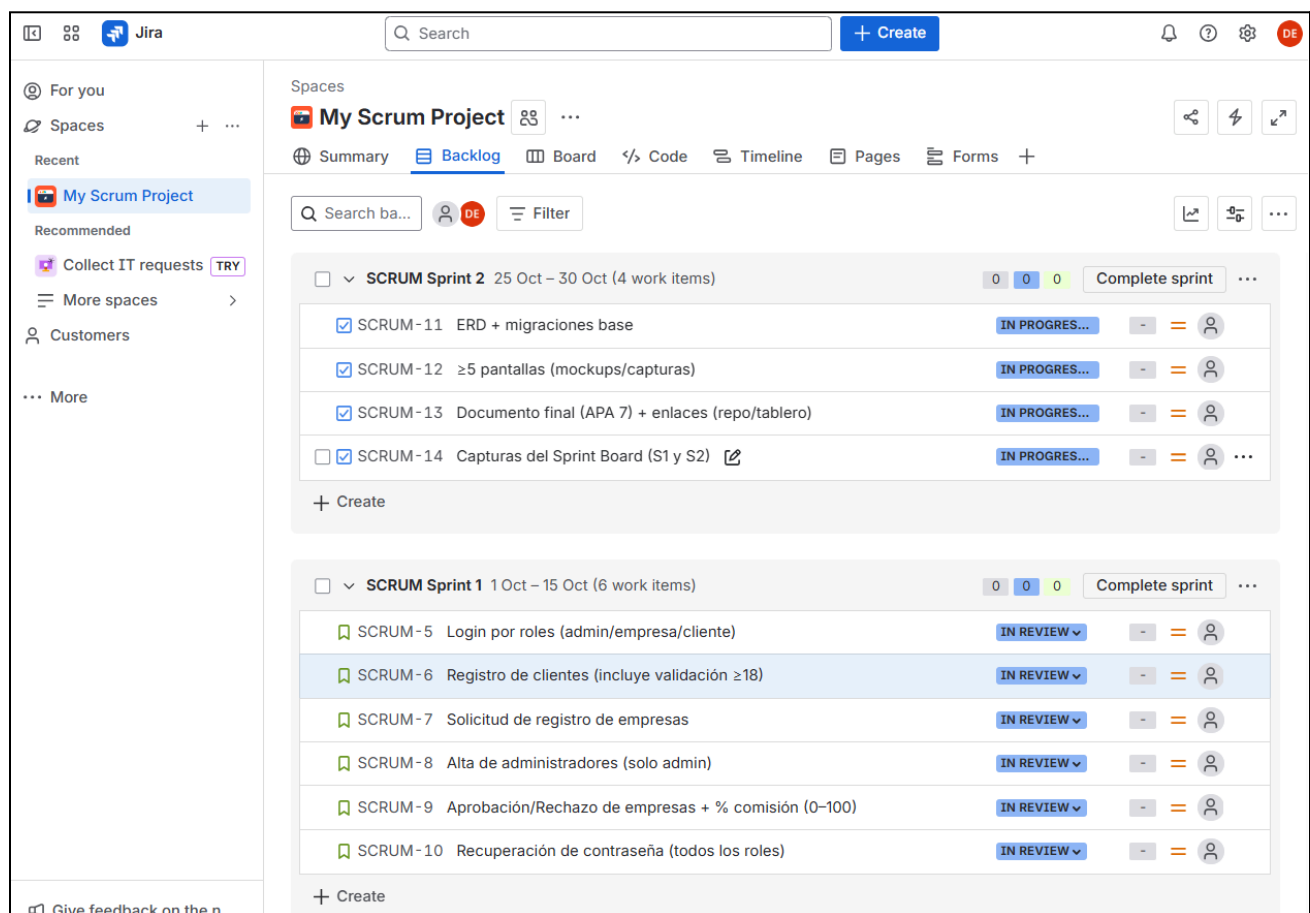
- **Daily (15 min):** cada integrante mueve su tarjeta y responde: qué hice / qué haré / bloqueos. Si Code Review/QA está lleno, todos ayudan a vaciarlo antes de empezar algo nuevo.
- **Working agreements:** nada pasa a Ready for Demo sin evidencia (captura o gif corto). Bugs encontrados regresan a In Progress con etiqueta bug.
- **Review:** se muestra el Incremento (funcional mínimo) con las capturas en el documento; el PO acepta/rechaza historias.
- **Retro:** ajustar WIP, pairing y estimaciones para el siguiente sprint (o resto de días).

4.8 Evidencia del Backlog en Jira

En la siguiente figura se muestra la **planificación del Backlog general del proyecto** dentro del entorno **Jira Software (metodología Scrum)**.

En esta vista se aprecian claramente los **dos sprints configurados**.

Figura 1. Backlog del proyecto en Jira (Sprint 1 y Sprint 2).



Fuente: Nota. Elaboración propia, 2025.

5. Modelo de datos

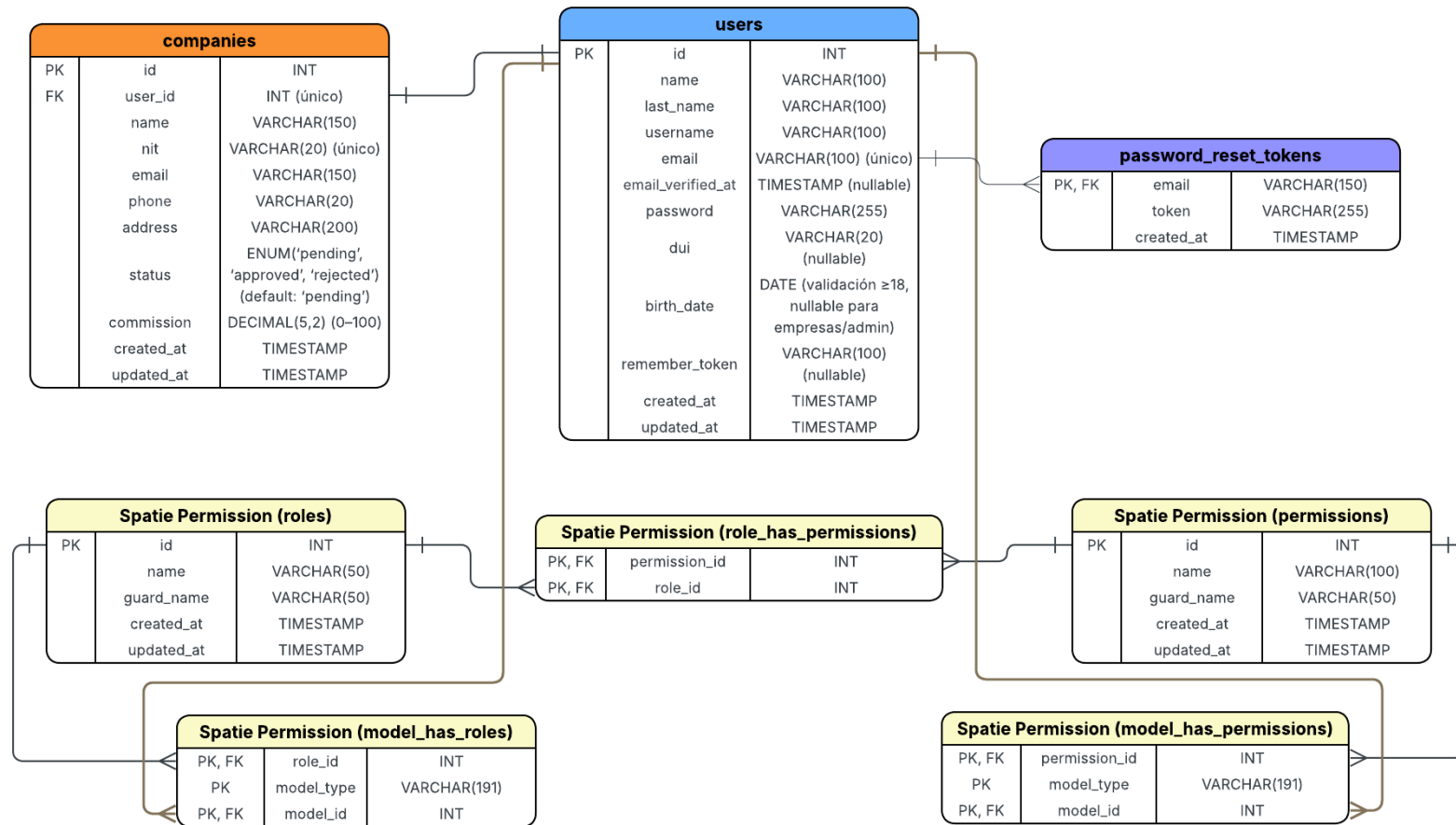
5.1 Diagrama entidad–relación (ERD)

El modelo de datos de “La Cuponera SV – Fase 1” está orientado a soportar los flujos de autenticación por roles, registro de clientes y empresas, recuperación de contraseñas y aprobación de empresas con porcentaje de comisión. El ERD contempla las entidades principales `users`, `companies` y `password_reset_tokens`, además del subsistema de autorización provisto por Spatie Permission mediante las tablas `roles`, `permissions` y sus tablas pivote (`role_has_permissions`, `model_has_roles`, `model_has_permissions`).

La relación `users–companies` es 1:1 (cada empresa tiene un usuario propietario) y la relación `users–password_reset_tokens` es 1:N (un usuario puede generar varios tokens). Las relaciones con Spatie se modelan como N:N mediante tablas pivote, lo que permite asignar múltiples roles y permisos a cada usuario y enlazar permisos a cada rol.

A continuación se presenta el diagrama entidad-relación (ERD).

Figura 2. Diagrama entidad relación del sistema “La Cuponera SV – Fase 1”.



Fuente: Nota. Elaboración propia, 2025

5.2 Diccionario de datos (estructura de tablas)

Tabla 12

Tabla users

Campo	Tipo	Clave	Descripción
id	INT	PK	Identificador del usuario
name	VARCHAR(100)		Nombre
last_name	VARCHAR(100)		Apellido (opcional)
username	VARCHAR(100)		Alias (opcional)
email	VARCHAR(100)	ÚNICO	Correo del usuario
email_verified_at	TIMESTAMP		Verificación de correo (nullable)
password	VARCHAR(255)		Contraseña cifrada
dui	VARCHAR(20)		Documento (opcional)
birth_date	DATE		Fecha de nacimiento (validación ≥ 18)
remember_token	VARCHAR(100)		Token “remember me”
created_at	TIMESTAMP		Fecha de creación
updated_at	TIMESTAMP		Última actualización

Fuente: *Nota*. Elaboración propia

Tabla 13*Tabla companies*

Campo	Tipo	Clave	Descripción
id	INT	PK	Identificador de empresa
user_id	INT	FK a users.id, ÚNICO	Usuario propietario (empresa)
name	VARCHAR(150)		Nombre comercial
nit	VARCHAR(20)	ÚNICO	Número de identificación tributaria
email	VARCHAR(150)		Email de contacto
phone	VARCHAR(20)		Teléfono
address	VARCHAR(200)		Dirección
status	ENUM('pending','approved','rejected')		Estado (por defecto pending)
commission	DECIMAL(5,2)		% comisión asignado (0–100)
created_at	TIMESTAMP		Fecha de creación
updated_at	TIMESTAMP		Última actualización

Fuente: *Nota*. Elaboración propia

Tabla 14*Tabla password_reset_tokens*

Campo	Tipo	Clave	Descripción
email	VARCHAR(150)	PK, FK a users.email	Correo asociado al token
token	VARCHAR(255)		Token de recuperación
created_at	TIMESTAMP		Fecha/hora de generación

Fuente: Nota. Elaboración propia

Tabla 15*Tabla roles (Spatie)*

Campo	Tipo	Clave	Descripción
id	INT	PK	Id del rol
name	VARCHAR(50)		admin, company, client
guard_name	VARCHAR(50)		Contexto (web)
created_at	TIMESTAMP		
updated_at	TIMESTAMP		

Fuente: Nota. Elaboración propia

Tabla 16*Tabla permissions (Spatie)*

Campo	Tipo	Clave	Descripción
id	INT	PK	Id del permiso
name	VARCHAR(100)		p. ej., approve_companies
guard_name	VARCHAR(50)		Contexto (web)
created_at	TIMESTAMP		
updated_at	TIMESTAMP		

Fuente: Nota. Elaboración propia

Tabla 17*Tabla role_has_permissions (Spatie)*

Campo	Tipo	Clave	Descripción
permission_id	INT	PK, FK a permissions.id	Permiso
role_id	INT	PK, FK a roles.id	Rol

Fuente: Nota. Elaboración propia

Tabla 18*Tabla model_has_roles (Spatie)*

Campo	Tipo	Clave	Descripción
role_id	INT	PK, FK a roles.id	Rol asignado
model_type	VARCHAR(191)	PK	Tipo de modelo (App\Models\User)
model_id	INT	PK, FK a users.id	Id del modelo

Fuente: Nota. Elaboración propia

Tabla 19*Tabla model_has_permissions (Spatie)*

Campo	Tipo	Clave	Descripción
permission_id	INT	PK, FK a permissions.id	Permiso asignado
model_type	VARCHAR(191)	PK	Tipo de modelo (App\Models\User)
model_id	INT	PK, FK a users.id	Id del modelo

Fuente: Nota. Elaboración propia

5.3 Reglas de integridad

5.3.1 Integridad de entidad

- Cada tabla posee PK única (users.id, companies.id, etc.).
- Campos obligatorios definidos según el diccionario (por ejemplo, users.email, companies.name).

5.3.2 Integridad referencial

- companies.user_id FK→users.id (1:1; índice único sobre user_id).
- password_reset_tokens.email FK a users.email (1:N).
- role_has_permissions.role_id FK a roles.id y role_has_permissions.permission_id FK a permissions.id.
- model_has_roles.model_id FK→users.id, model_has_roles.role_id FK a roles.id.
- model_has_permissions.model_id FK a users.id, model_has_permissions.permission_id FK a permissions.id.
- Políticas sugeridas: ON DELETE RESTRICT en companies.user_id; ON DELETE CASCADE en pivotes.

5.3.3 Integridad de dominio

- users.email con restricción UNIQUE y formato de correo válido.
- companies.nit UNIQUE.
- companies.status \in {pending, approved, rejected} (default pending).
- companies.commission en rango [0, 100].
- users.birth_date válida edad ≥ 18 para clientes.

5.3.4 Reglas de negocio y seguridad

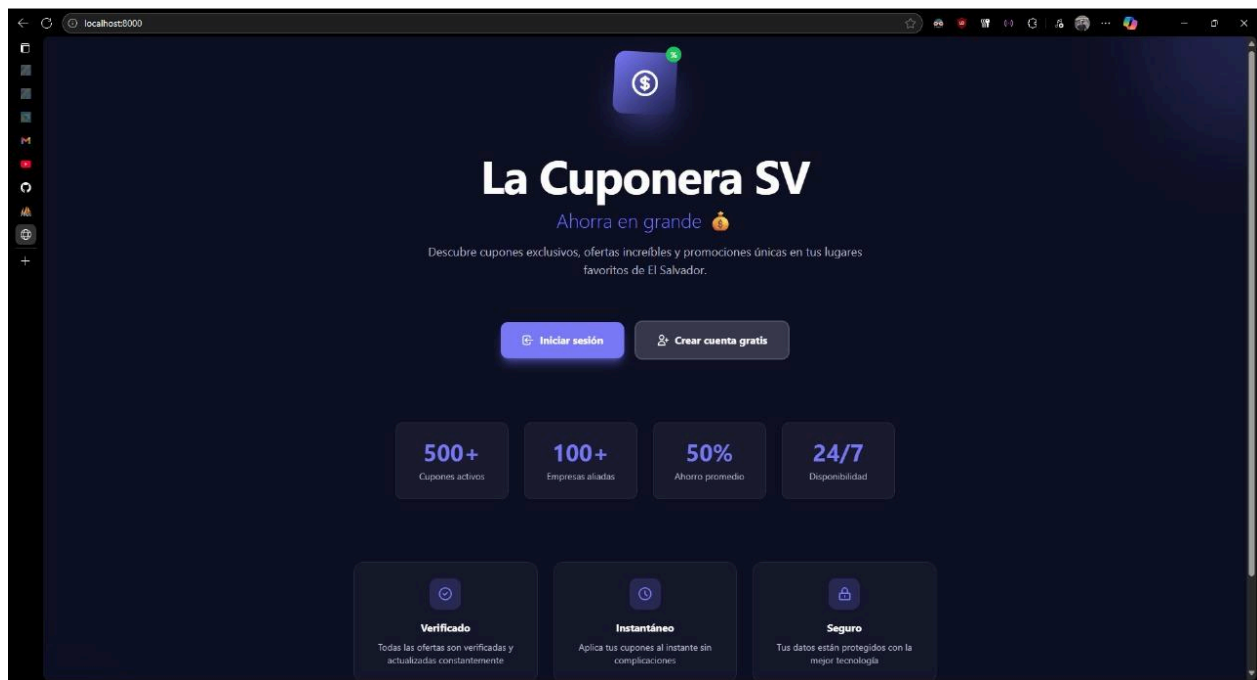
- Solo los administradores pueden aprobar/rechazar empresas y definir la comisión..
- Contraseñas en users.password se almacenan hasheadas; protección CSRF en formularios.
- Tokens de password_reset_tokens expiran y se invalidan tras su uso.
- Acceso a módulos controlado por roles y permisos (Spatie).

6. Diseño de interfaces

En esta sección se presentan las principales pantallas del sistema “La Cuponera SV”, desarrolladas como parte del mínimo funcional de la Fase #1.

Las interfaces fueron diseñadas con un enfoque intuitivo, moderno y responsive, utilizando una paleta oscura y elementos visuales consistentes que refuerzan la identidad del proyecto.

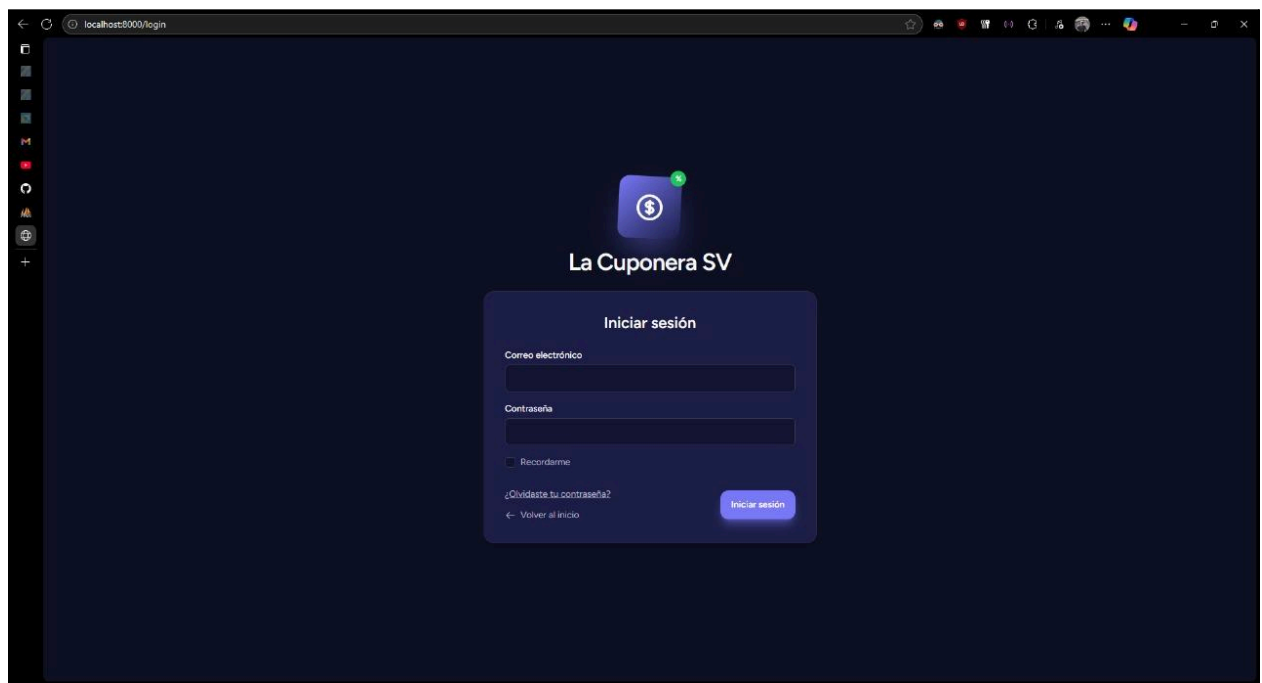
Figura 3. Página principal (Home) del sitio “La Cuponera SV”.



Fuente: Nota. *Elaboración propia, 2025.*

La interfaz muestra el mensaje de bienvenida, botones de acceso (“Iniciar sesión” y “Crear cuenta gratis”) y estadísticas visuales del sitio (cupones activos, empresas aliadas, ahorro promedio y disponibilidad). Permite a cualquier visitante explorar promociones sin iniciar sesión.

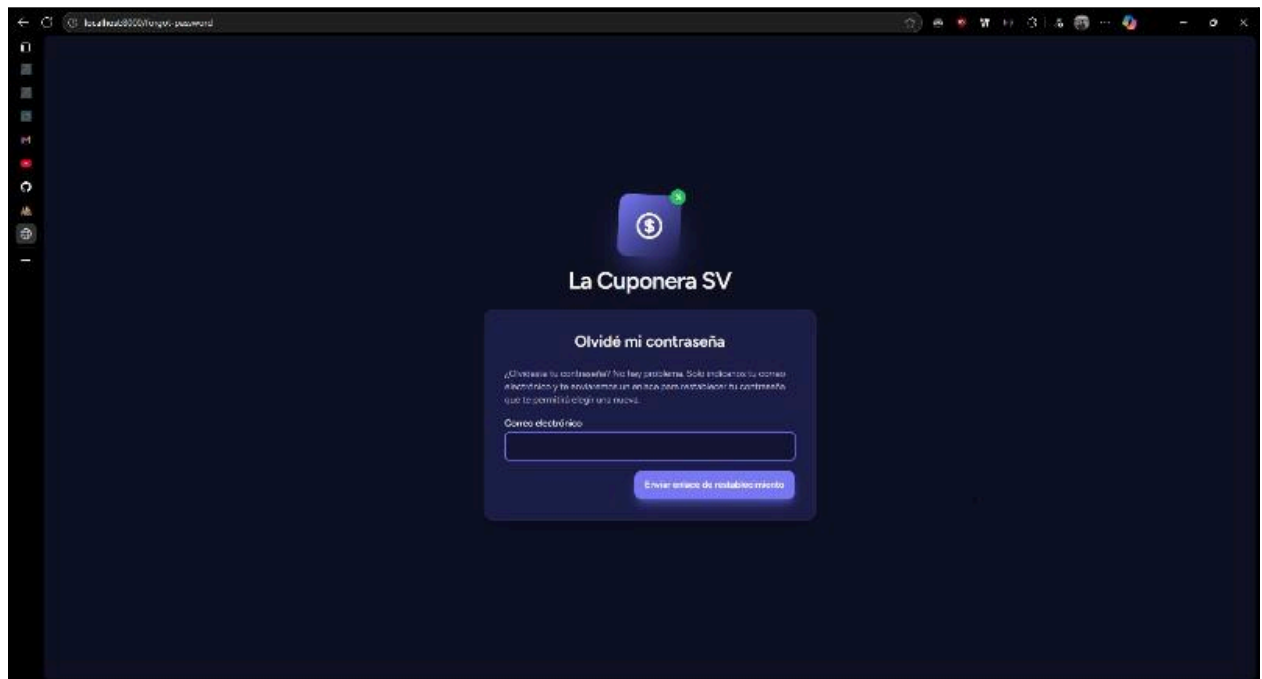
Figura 4. Pantalla de inicio de sesión.



Fuente: *Nota. Elaboración propia, 2025.*

Vista del formulario para ingresar con correo electrónico y contraseña. Incluye la opción de recordar sesión y un enlace directo para recuperar la contraseña. Esta pantalla forma parte del flujo de autenticación para los tres tipos de usuarios (administrador, empresa y cliente).

Figura 5. Pantalla de recuperación de contraseña.



Fuente: Nota. *Elaboración propia, 2025.*

Formulario que solicita la dirección de correo electrónico del usuario y envía un enlace de restablecimiento. Implementa validaciones para evitar correos inexistentes o vacíos, cumpliendo con el requisito funcional RF-12.

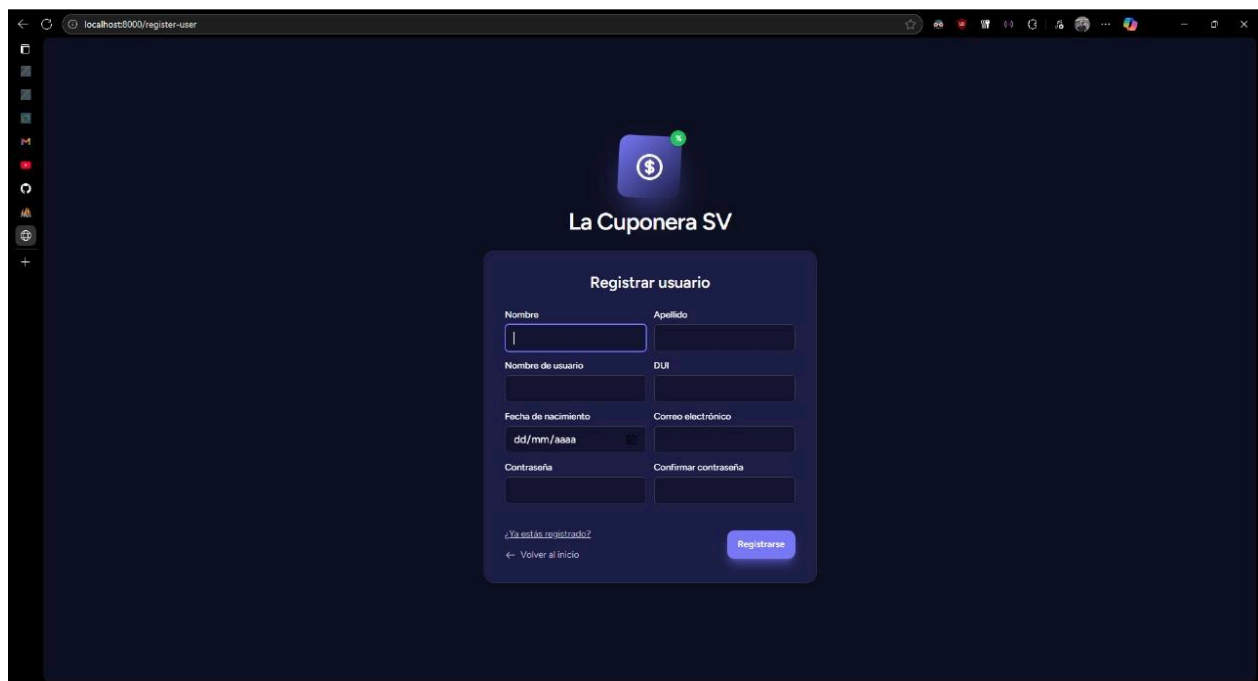
Figura 6. Formulario de registro de empresas.

The image shows a web browser window with the address bar displaying 'localhost:8000/register-company'. The page has a dark blue background. At the top center is a logo consisting of a blue square with a white dollar sign and a green checkmark, with the text 'La Cuponera SV' below it. Below the logo is a white-bordered box titled 'Registrar compañía'. Inside this box, there is a welcome message: '¡Bienvenido al registro de empresas! Completa el formulario a continuación para crear una cuenta de empresa y comenzar a atraer más clientes a tu negocio.' Below the message are several input fields: 'Nombre de la compañía' and 'NIT' (each with a small validation icon), 'Teléfono' and 'Correo electrónico', and a larger 'Dirección' field. At the bottom of the form are 'Contraseña' and 'Confirmar contraseña' fields. Below these fields is a disclaimer: 'Al registrarte, aceptas nuestros Términos y Condiciones y Política de Privacidad. Tu empresa estará sujeta a un proceso de verificación antes de activar la cuenta.' At the very bottom of the form box are two links: '¿Ya estás registrado?' and 'Volver al inicio', and a blue 'Registrarse' button.

Fuente: Nota. *Elaboración propia, 2025.*

Interfaz donde las compañías completan su solicitud de registro. Incluye validación de datos como NIT, correo, dirección y contraseña, quedando la solicitud pendiente de aprobación del administrador.

Figura 7. Formulario de registro de usuarios (clientes).

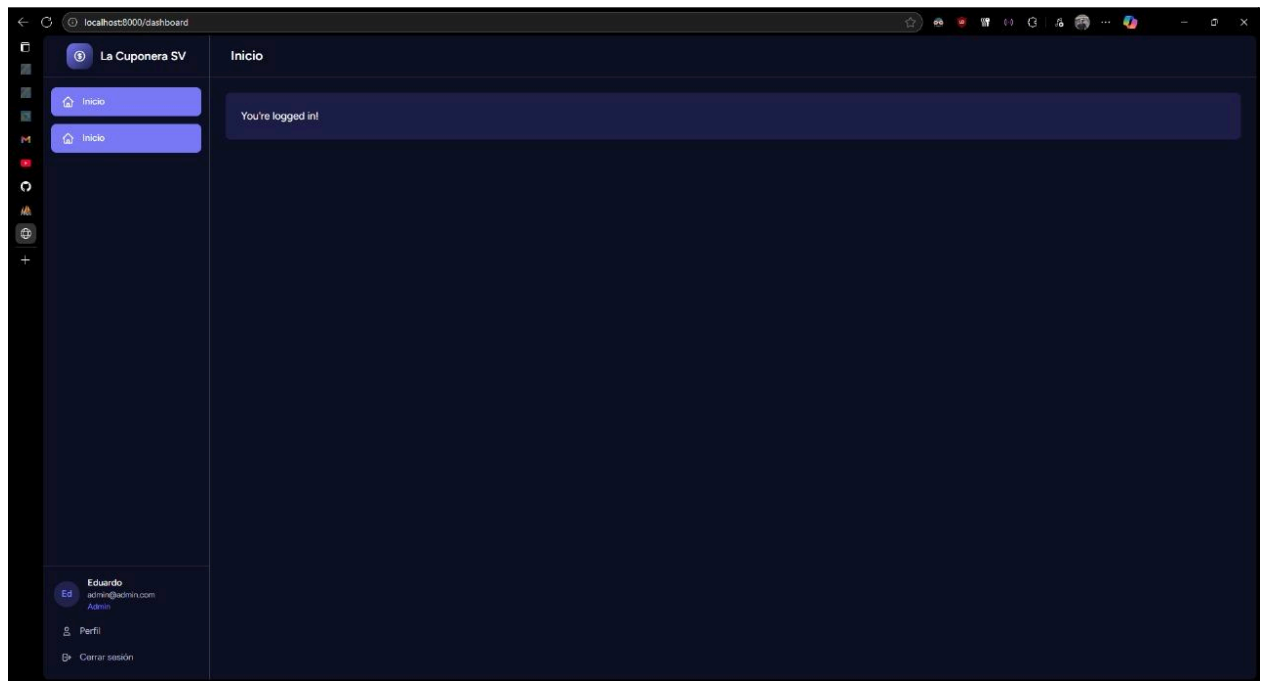


The image shows a web browser window with the address bar displaying 'localhost:8000/register-user'. The page has a dark blue background. At the top center, there is a logo consisting of a blue square with a white dollar sign and a green circle above it, followed by the text 'La Cuponera SV'. Below the logo is a white-bordered box titled 'Registrar usuario'. Inside this box, there are several input fields arranged in two columns. The first column contains fields for 'Nombre', 'Nombre de usuario', 'Fecha de nacimiento' (with a date picker showing 'dd/mm/aaaa'), and 'Contraseña'. The second column contains fields for 'Apellido', 'DUI', 'Correo electrónico', and 'Confirmar contraseña'. At the bottom left of the box, there is a link that says '¿Ya estás registrado?' and a button that says '← Volver al inicio'. At the bottom right, there is a blue button that says 'Registrarse'.

Fuente: Nota. *Elaboración propia, 2025.*

Pantalla que permite crear una cuenta de cliente. Integra validación de edad mínima (≥ 18 años), verificación de formato de DUI y confirmación de contraseñas, cumpliendo con el RF-02.

Figura 8. Panel principal del administrador.



Fuente: *Nota. Elaboración propia, 2025.*

Vista posterior al inicio de sesión del rol **administrador**, donde se confirma el acceso exitoso al sistema y se muestran los elementos del menú lateral: inicio, perfil y cierre de sesión. Este panel servirá de base para la gestión de empresas y usuarios en la siguiente fase.

7. Repositorio y control de versiones

El control de versiones del proyecto “**La Cuponera SV – Fase 1**” se gestionó mediante la plataforma **GitHub**, utilizando el repositorio oficial:

https://github.com/eduardoezequiel/cuponera_LIS_2025

Este repositorio almacena el código fuente completo del sistema desarrollado en **Laravel**, incluyendo controladores, vistas Blade, migraciones, seeders y archivos de configuración. Además, contiene un archivo **README.md** con las instrucciones detalladas para la instalación, configuración de la base de datos y ejecución local del proyecto.

8. Conclusiones y próximos pasos

Durante la Fase 1 del proyecto “La Cuponera SV”, se logró establecer la estructura base del sistema, implementando los módulos esenciales de autenticación, registro y recuperación de contraseñas para los tres tipos de usuario: administrador, empresa y cliente.

El trabajo se desarrolló bajo la metodología Scrum, utilizando Jira para la gestión del backlog y la planificación de sprints, lo que permitió una distribución eficiente de tareas y una comunicación constante entre los integrantes.

Se consolidó un entorno técnico sólido con Laravel, Blade y TailwindCSS, aplicando control de versiones en GitHub y manteniendo buenas prácticas de documentación a través del archivo README.md, que detalla la instalación, configuración y ejecución del proyecto.

En esta etapa se cumplió con el alcance establecido en la rúbrica, asegurando un producto funcional y documentado.

Próximos pasos

Para la siguiente fase se plantea:

- Implementar la **gestión completa de cupones y ofertas**.
- Simular el **proceso de compra y facturación**.
- Incorporar **reportes administrativos** sobre ventas y ganancias.
- Ampliar el diseño responsivo y mejorar la experiencia del usuario.
- Integrar **pruebas automáticas** que refuercen la calidad del sistema.

Estos avances permitirán consolidar a “La Cuponera SV” como una plataforma web funcional, escalable y orientada al fortalecimiento de las microempresas salvadoreñas.

9. Referencias

- Atlassian. (2025). *Jira Software – Guía para tableros Scrum*. Recuperado de <https://www.atlassian.com/es/software/jira>
- Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., ... Thomas, D. (2001). *Manifesto for Agile Software Development*. Recuperado de <https://agilemanifesto.org/iso/es/manifesto.html>
- GitHub. (2025). *cuponera_LIS_2025 [Repositorio de código]*. Recuperado de https://github.com/eduardoezequiel/cuponera_LIS_2025
- Laravel. (2025). *Laravel Documentation*. Recuperado de <https://laravel.com/docs>
- Microsoft. (2024). *Git y control de versiones en proyectos colaborativos*. Recuperado de <https://learn.microsoft.com/es-es/devops/develop/git/>
- OpenAI. (2025). *Buenas prácticas de documentación técnica para proyectos académicos*. Recuperado de <https://openai.com/research>
- Pressman, R. S., & Maxim, B. R. (2020). *Ingeniería de software: un enfoque práctico* (8.^a ed.). México: McGraw-Hill.
- Sommerville, I. (2019). *Ingeniería del software* (10.^a ed.). Madrid: Pearson Educación.
- Tailwind Labs. (2025). *Tailwind CSS Documentation*. Recuperado de <https://tailwindcss.com/docs>
- Universidad Don Bosco. (2025). *UIA1 – Lenguajes Interpretados en el Servidor (LIS941): Unidad 1*. San Salvador: UDB Virtual.
- Universidad Don Bosco. (2025). *UIA2 – Lenguajes Interpretados en el Servidor (LIS941): Unidad 2*. San Salvador: UDB Virtual.
- Universidad Don Bosco. (2025). *UIA3 – Lenguajes Interpretados en el Servidor (LIS941): Unidad 3*. San Salvador: UDB Virtual.

- Universidad Don Bosco. (2025). *UIA4 – Lenguajes Interpretados en el Servidor (LIS941): Unidad 4*. San Salvador: UDB Virtual.
- Universidad Don Bosco. (2025). *Ruta de aprendizaje – LIS941 – C02 – 2025*. San Salvador: UDB Virtual.
- W3Schools. (2025). *HTML, CSS & JavaScript Tutorials*. Recuperado de <https://www.w3schools.com/>